The Past, Present, and Future of Ground Station Automation within the DSN

Forest Fisher, Darren Mutz, Tara Estlin, Leslie Paal, Emily Law, Nasser Golshan, Steve Chien
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive, M/S 126-347
Pasadena, CA 91109-8099
818-393-5368
Forest.Fisher@jpl.nasa.gov

Abstract—This paper describes an architecture for an autonomous Deep Space Tracking Station (DS-T). The architecture targets fully automated routine operations encompassing scheduling and resource allocation, antenna and receiver predict generation, track procedure generation from service requests, and closed loop control and error recovery for the station subsystems. This architecture has been validated by the construction of a prototype DS-T station, which has performed a series of demonstrations of autonomous ground station control for downlink services with NASA's Mars Global Surveyor.

1. INTRODUCTION

The Deep Space Network (DSN) was established in 1958 and has since evolved into the largest and most sensitive scientific telecommunications and radio navigation network in the world. The purpose of the DSN is to support unmanned interplanetary spacecraft missions and to support radio and radar astronomy observations taken in the exploration of space. The function of the DSN is to receive telemetry signals from spacecraft, transmit commands that control spacecraft operating modes, generate the radio navigation data used to locate and guide a spacecraft to its destination, and acquire flight radio science, radio and radar astronomy, very long baseline interferometry (VLBI), and geodynamics measurements.

This paper describes the Deep Space Terminal (DS-T), a prototype 34-meter deep space communications station developed as a technology demonstration of *fully autonomous, lights-out*, operations. In the DS-T concept, a global DSN schedule is disseminated to a set of autonomous DS-T stations. Each DS-T station operates autonomously, performing tracks in a largely independent fashion. When requested to perform a track, the DS-T station performs a number of tasks (at appropriate times) required to execute the track. First, the DS-T station uses appropriate spacecraft navigation ephemeris and predict generation software in order to produce necessary antenna and receiver predict information required to perform the track. Next, the DS-T station executes the pre-calibration process, in which the antenna and appropriate subsystems (e.g., receiver, exciter, telemetry processor, etc.) are configured in anticipation of the track. During the actual track, the signal from the spacecraft must be acquired and the antenna and subsystems must be commanded to retain the signal, adjust for changes in the signal (such as changes in bit rate or modulation index as transmitted by the spacecraft), and perform error recovery. Finally, at the completion of the track, the station must be returned to an appropriate standby state in preparation for the next track. All of these activities require significant automation and robust execution including closed loop control, retries and contingency handling.

In order to provide this autonomous operations capability, the DS-T station employs tightly coupled state of the art hardware and software. The DS-T architecture encompasses two levels: the network level and the station level. Within this paper we focus primarily on the station level, but also describe the aspects of the network layer as relevant to the integration of the DS-T into the overall Deep Space Network architecture.

The network layer represents the Deep Space Network wide operations capability necessary to determine the DS-T operations activities over a medium range time scale (a weekly basis) at a high level of activity (the services the DS-T station is to provide to spacecraft over each specific period of time during the week).

Within the DS-T station itself, there are three layers within the software and hardware: the DS-T automation layer, the DS-T application layer, and the DS-T subsystem layer. First, at the network layer the JPL scheduler layer accepts track requests (along with service definitions: downlink, uplink, uplink/downlink, etc.) from the flight projects and produces a local schedule for each DS-T station. Second, the DS-T automation layer resides locally at the DS-T site and accepts a local schedule from the scheduler layer. This schedule is interpreted by a schedule executive which, for each track, causes track script generation to execute and causes execution of the track script itself. The final component of the DS-T automation layer is the Downlink Monitor, which runs the scripts that perform the actions for each specific track. The Downlink Monitor is also part of the DS-T application layer where it interfaces to the subsystems.

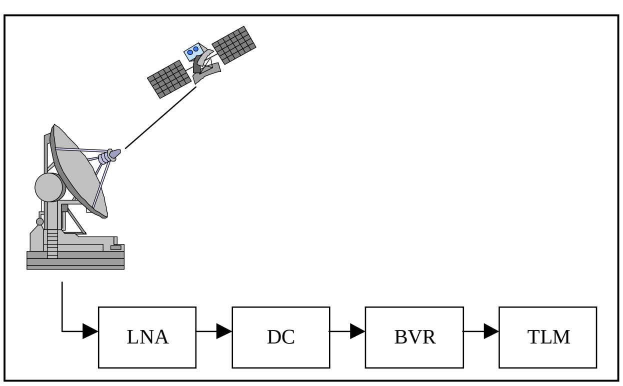
In May 1998, the DS-T prototype first demonstrated automated downlink capability of single isolated tracks for the Mars Global Surveyor (MGS) spacecraft. Since May, many multi-day demonstrations have taken place including a six day unattended demonstration. During these demonstrations, a service request for downlink services, a track sequence of events, and spacecraft ephemeris were used to automatically downlink data from the MGS spacecraft. Future demonstrations of the DS-T prototype include: autonomous downlink tracking of the New Millennium Deep Space One (NM DS1) Spacecraft, and support of the DS1 Beacon Monitor Experiment.

Included in NM DS1 support is support of the Beacon Monitor Experiment (BMOX), in which the spacecraft will initiate a track request by communicating a low bandwidth signal to a small antenna which will automatically trigger the scheduling of a demand access track and subsequent automated execution of the track at the DS-T station. For the initial BMOX demonstration the DS-T station will be used to perform tone detection as well.

In the remainder of the paper we describe the overall architecture and how it fits into the DSN operations architecture. First we describe each of the layers in the DS-T architecture: the network layer, and the layers comprising an individual station layer (the automation layer, protocol layer, and subsystem layer.) We then describe in further detail the current status of the implementation of the architecture presented, discuss our results, talk about future work, and finally we make comparisons to other systems.

2. Signal Data Flow

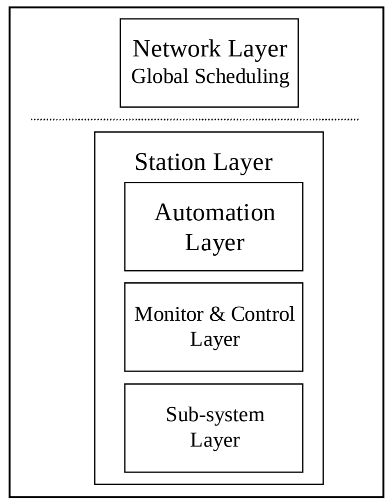
The hardware architecture for the Deep Space Terminal (DS-T) ground communication station primarily consist of a 34m beam wave guide antenna, a low noise amplifier (LNA), a down converter (DC), a Block V Receiver (BVR), and a telemetry (TLM) processor (see Figure below).



In order to create a communication link with a spacecraft, each of these components must be configured and networked together. The spacecraft involved with the link affects the configuration, while different configurations are required for different track services. In the case of DS-T, these services consisted of different forms of downlink. In the case of a DS-1 Beacon track, a different receiver (FSR) is used due to the low bandwidth signal.

3. Overall Architecture

The DS-T uses a layered architecture approach (see Figure below). The lowest layer is made up of the sub-systems themselves. In conjunction with the hardware mentioned above additional sub-systems make up the lowest layer of the architecture, the sub-system layer. These additional sub-systems include controllers for the components already mentioned and sensory sub-systems,



Layered above the hardware or sub-system layer is the application layer. This layer is comprised of the control software used in commanding the sub-systems in the hardware layer and monitoring the status of these systems.

The top layer is the automation layer, which comprises all of the automation software. The automation layer also provides the interface to the autonomous station/terminal. It is through the automation layer that service requests are submitted to the system and then scheduled for execution.

The layered approach to the architecture provides several benefits. First and foremost it provides well-defined boundaries between different functionalities within the system. This combined with the intuitive abstraction levels makes decomposition of the system simpler, aiding in division of development and testing responsibilities.

4. Network Architecture

When the decision is made to fly a mission, a very knowledge-intensive process begins that will ensure the necessary

DSN antenna coverage. First, a forecast is made of the DSN resources that the spacecraft will require. In the Resource Allocation Process (RAP), the types of services, frequency, and duration of the required tracks are determined as well as high-level resource requirements (e.g., antenna). While the exact timing of the tracks is not known, a set of automated forecasting tools are used to estimate network load and to assist in ensuring that adequate network resources will be available. One part of the network architecture is a unified tool suite that has been developed called TMOD Integrated Ground Resource Allocation System (TIGRAS), which uses operations research and probabilistic reasoning techniques to allow forecasting and capacity planning for DSN resources [4].

As the time of the actual tracks approaches, this estimate of resource loading is converted to an actual schedule, which becomes more concrete as time progresses. In this process, specific project service requests and priorities are matched up with available resources in order to meet communications needs for earth-orbiting and deep space spacecraft. This scheduling process involves considerations of thousands of possible tracks, tens of projects, tens of antenna resources and considerations of hundreds of subsystem configurations. In addition to adding the detail of antenna subsystem allocation, the initial schedule undergoes continual modification due to changing project needs, equipment availability, and weather considerations. Responding to changing context and minimizing disruption while rescheduling is a key issue.

An evolution of the Operation Mission Planner (OMP-26M[1]) system, the Demand Access Network Scheduler (DANS) [6] is designed to deal with the more complex subsystem and priority schemes required to schedule the larger 34 and 70 meter antennas. Because of the size and complexity of the rescheduling task, manual scheduling is prohibitively expensive. Automation of these scheduling functions is projected to save millions of dollars per year in DSN operations costs.

DANS uses priority-driven, best-first, constraint-based search and iterative optimization techniques to perform priority-based rescheduling in response to changing network demand. In these techniques, DANS first considers the antenna allocation process, as antennas are the central focus of resource contention. After establishing a range of antenna options, DANS then considers allocation of the 5-13 subsystems per track (out of the tens of shared subsystems at each antenna complex) used by each track.

The network layer has three principle interfaces to lower levels in the automation architecture. In addition to resource allocation, the network layer is responsible for storing information on the tracking services required by the spacecraft, current spacecraft configuration, planetary and spacecraft ephemeris, and telecommunications models. This information (as well as the current schedule) is stored in a globally accessible database called the Mission and Assets Database (MADB). The MADB is a major interface point from the network layer to the automation element of the station layer.

Another required capability of the DSN and the DS-T network layer is to generate near real time telemetry and monitor data as well as performance summarizations. These are generated by the monitor and control layer of the DS-T station and are forwarded on to the network layer for appropriate distribution.

5. Station Architecture

The station layer repesents the actual hardware and software dedicated to a single DS-T station. There are three principal components to the station layer: the automation layer, the monitor and control layer, and the subsystem layer. The automation layer is responsible for the high level control and execution monitoring of the DS-T station. The monitor and control layer is responsible for low level control of the antenna track as well as logging and archiving relevant monitor data. The subsystem level provides a uniform interface to the antenna subsystems to facilitate modular software design and reduce the effort needed to interchange and upgrade hardware.

The Automation Layer

The automation layer performs several functions within the DS-T UNIX workstation, all relating to automation and high level monitor and control for the DS-T station. This layer consists of six components: the service request processor, the schedule executive, configuration engine, predict generators, script generator [9], and the station controller.

The DS-T's *service request* (SR) processor takes in a service request and generates first cut configuration files needed to produce a scheduling request and define the request to be performed. From these files the request is passed to the DANS scheduling system to produce the network schedule.

The *schedule executive* (SE) takes the network schedule and sets up the station schedule for execution and provides the means for automated re-scheduling and/or manual schedule editing in the event of changes to the master schedule. Schedule execution is set up by parsing the schedule and scheduling the sub-tasks which need to be performed in order to accomplish the originally scheduled activity. Each subtask is placed into a UNIX crontab file with the appropriate time stamp, relative to the Aquisition Of Signal (AOS). In this manner, each of the remaining components of the automation layer are invoked at the appropriate time by the UNIX crontab facility.

The *configuration engine* (CE) is the first to be started up by the cron facility. This component is responsible for retrieving all the necessary data/data files needed for station

[1] Scheduling tool for 26-meter antenna network.

operations, from a collection of data stores. These files contain information about: spacecraft trajectory, needed to calculate antenna pointing predicts; spacecraft view periods (when the spacecraft is visible to the antenna); models of planetary orbits, to determine if the spacecraft view is obstructed; precise location of the ground station; and activity service packages (ASP). The ASPs contain the service request which define the type of activity desired by a mission/project and activity details like carrier frequency, symbol rate, and project mission profiles. The CE examines this vast collection of data and extracts the relevant information into configuration files for the remaining modules of the automation layer.

After the CE creates the needed configuration files for the *predict generators* (PG) and the *script generator* (SG), the cron facility invokes the SG processes with its appropriate configuration files.

The SG is where the majority of the control autonomy comes from. The SG uses Artificial Intelligence Planning techniques to perform a complex software module reconfiguration process [7]. This process consists of piecing together numerous highly interdependent smaller control scripts in order to produce a single script to control the operations of the DS-T station.

The core engine used in the SG is the Automated Scheduling and Planning ENvironment (ASPEN) [10]. The ASPEN system is a reusable, configurable, generic planning/ scheduling application framework that can be tailored to specific domains to create conflict-free plans or schedules. It has a number of useful features including an expressive modeling language, a constraint management system for representing and maintaining antenna operability and/or resource constraints, a temporal reasoning system and a graphical interface for visualizing plans and states. ASPEN has been adapted to input antenna tracking goals and automatically produce the required command sequence necessary to create the requested link.

The control script produced by the SG: sets up the track by configuring the station during pre-track; provide the track service requested by commanding the antenna and subsystems to acquire and maintain lock on the signal throughout mode changes; and cleanup and shutdown the station at the completion of the track.

It is during the pre-track that the predict generation (PG) process takes place. The PG functionality consists of three predict generators used to calculate: antenna pointing predicts (AP-PDX), radiometric predicts (RAD-PDX), and telemetry predicts (TEL-PDX). Another requirement of the DS-T was to provide the means of generating on station PDXs or to use provided PDXs. This show another example of how the DS-T SG reconfigures the pre-track, by selecting which predicts (PDXs) are tobe generated.

As previously mentioned, the station controller (SC) spans both The Automation Layer and The Station Monitor and Control Layer. As such the explanation of the SC functionality is left for the next section of this paper.

The Station Monitor and Control Layer

The Station Monitor and Control process acts as an agent for the Automation Layer, executing the generated scripts. The Monitor and Control (M&C) layer expands the high level directives of the script into subsystem dependent directives, isolating the automation layer from the lower levels. By using the monitor information from the Station Monitor process, the script execution path is altered as necessary to accommodate external events.

All subsystem generated monitor information (monitor data packets and event notices) is processed in the Station Monitor process. The monitor data is recorded in a data store and condensed performance reports are generated for the higher level processes.

The Uplink/Downlink process handles the spacecraft command and telemetry data flow. The command data is accepted as Command Link Transmission Units (CLTUs) or as command packet files and processed according to Consultative Committee for Space Data Systems (CCSDS) standards. Telemetry data is formatted in the subsystem into frames or packets. These are archived until the data is delivered to the mission or the Product Data Deliver System (PDDS).

For debugging and experimental use the M&C layer has the capability to handle low level directives for the subsystems in bypass mode.

The Station Subsystem Layer

The Subsystem interface layer handles all communication protocol and connection related work. This is necessary because the DS-T is a mix of COTS (commercial off the shelf) and custom JPL designed equipment using a variety of protocols. The inherited JPL equipment uses a proprietary communication protocol, while some COTS units use TCP/IP, and others use either the IEEE-488 or RS-232 low level protocols. The JPL protocol also requires the equipment “to be assigned” to a track, requiring some hereditary connection management.

6. DESIGN CRITERIA

The original goal was to build an autonomous control system for a deep space communications station. This system had to meet the following criteria: schedule driven with a high level service request interface; an automated scheduling component for initial scheduling and rescheduling; provide script guided control; ability to generate predicts or use provided predicts; automatically configure pre-pass; utilization of COTS components wherever feasible; operations based on defined but expandable set of services; autonomous error recovery for a defined class of problems;

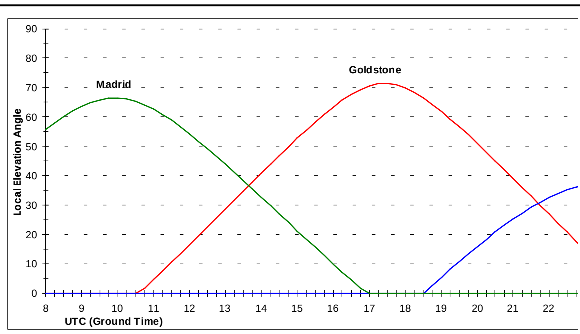
post pass data delivery; and treat ground terminal as a network computer with an RF peripheral.

Although each of these criteria where met, in an effort to save space we will focus on a few key elements. One of the most important points was the idea of a ground station looking just like a network computer to a user, operator, or mission. This is best demonstrated by an operational scenario. To provide service a user need only login to the DS-T workstation and submit a service request to the scheduling system, or FTP a schedule and service request to a particular file system location. From either of these inputs DS-T would detect the existence of a track/service schedule and proceed to schedule station specific task and configure the station to provide the service and finally when the time comes the track would begin without further user interaction.

As mentioned above, the station reacts to a service-request-derived schedule generated by an automated scheduling system. It is through the reaction to this schedule that the dynamic track specific control scripts are generated. It is through the execution of these control scripts that the autonomous operations of the station takes place. And finally as for the COTS systems, both the monitor and control software that interprets the control script and the telemetry processor were commercially provided.

7. RESULTS

The most important evaluation of a system is *does it work*. In order to provide qualitative results, the DS-T was demonstrated through a series of one to six day demonstrations. For September 16, 1998, a representative day during our six day autonomous unattended demonstration, we collected above 90% of the transmitted frames. This performance is on par with the operator-controlled stations.



Track Segments:

#1	11:15:00 – 14:00:23	3way/65	(Madrid)
#2	14:00:23 – 15:42:26	LOS	
#3	15:42:26 – 15:52:23	1way	
#4	15:42:26 – 15:52:23	3way/25	(Goldstone)
#5	15:52:23 – 20:03:34	1way	
#6	20:03:34 – 23:40:00	3way/34	(Canberra)

In the figure above, the graph represents when MGS was in view of the ground stations at each of the three complexes (Madrid, Goldstone, and Canberra). DS-T, which is located at Goldstone, tracked MGS through the five track segments indicated in the figure. Track segment 2, which is labeled LOS, indicates that there was a scheduled loss of signal (LOS) so during this segment no frames were collected. During each of the other respective track segment DS-T collected 75%, 91%, 96%, 90%, 23% of the broadcasted frames. As shown by the graph, during segment 1 and 6 the elevation of the dish is low in the sky. Under these circumstances there is considerably more atmospheric interference which explains the lower percent of frame collection. On the other hand, if you look at segment 4 where there is a long segment with the spacecraft high in the sky the data collection is quite high. In segment 3 and 5 the values are a little lower due to the shortness of the segments. This is explained by the fact that some data is lost during a change in mode, as in the transition from LOS to 1way and 3way/25 to 1way.

We will take just a moment to explain the modes during this track. When a spacecraft is downlinking data it is said to be in 1way mode. When an uplink and a downlink are taking place simultaneously the spacecraft is said to be in 2way mode. If a station is communicating in 2way mode with a spacecraft, and another station is listening in on the downlink of the spacecraft, the second station is said to be in 3way with the 2way station. Because DS-T is not equipped for uplink , DS-T operates in either 1way or 3way mode. In this case during segment 4 dss25 (deep space station) was in 2way and DS-T was in 3way with 25 (3way/25). Because the downlink frequency is relative to the uplink frequency, it is critical to determine the station involved in the uplink.

8. FUTURE WORK

Building on the success and knowledge gained from the DS-T automation software, we have on going work to provide dynamic commanding. This approach will integrate the planning aspect of control with the monitor and control component. By integrating these two aspects of automation, a monitor and control system is able to provide more thorough error recovery with reduced reaction time. This work is being done on a system called CLEaR (Closed Loop Error Recovery). The CLEaR system is also being integrated with a fault detection system (FDIR – Fault Detection, Isolation and Recovery) to provide intelligent analysis of monitor data. Once the intelligent analysis is performed CLEaR will reason about the diagnostics and provide an intelligent response. Both CLEaR and FDIR are explained in more detail in the Insights to DSN Automation section of this paper.

9. COMPARISON TO OTHER WORK

There are a number of existing systems, which integrate scheduling, planning, control, and execution monitoring. We do not attempt to review them all, but focus on a few representative systems. To begin with, the main distinction between this architecture and other work is the hierarchical structure and the complexity of the DSN antenna operations domain.

Brooks' subsumption architecture [5] contains no hierarchy of planning, scheduling, or control. This type of architecture has been used for mobile robot navigation, where replanning and rescheduling is a more constrained problem as compared to antenna operations which must schedule and plan for multiple resources (antennas and subsystems), and with both hard and soft temporal constraints.

CIRCA [17] has a three-tiered architecture comprised of a planner, scheduler, and an executor that interacts with the environment through actuators and sensors in a mobile robot navigation domain. CIRCA does planning then scheduling, versus the DSN automation architecture which must first schedule and then plan. CIRCA's scheduling enforces hard real-time constraints, but returns failure if it cannot meet the time constraints. DANS/OMP, on the other hand, enforces hard real-time constraints, but always returns a schedule, by using the priority scheme, which maximizes the number of project requests that it accommodates. If some project requests cannot be accommodated, DANS/OMP will still return a schedule, even though it is sub-optimal.

$_3$T [3] is a three-tiered architecture with a planner, sequencer, and a reactive skill module that interacts with the environment. Planning occurs hierarchically before sequencing, unlike the architecture, which we describe in this paper, which does scheduling then planning. The sequencer in $_3$T is a RAP [11] interpreter that encodes all the timing information within the RAPs. DANS/OMP does not use RAPs, and uses a more complex algorithm to schedule the projects' requests. Unlike the DSN automation architecture, in $_3$T all three of its tiers do not need to be used for a given task. In the DSN domain necessarily scheduling, then planning, then control and execution must happen for successful antenna operations.

ATLANTIS [12] is also a three-tiered architecture, similar to $_3$T. It is comprised of a controller which acts at the lowest reactive level, a sequencer which is a special-purpose operating system based on the RAP system, and a deliberator which does planning and world modeling. In ATLANTIS, it is the sequencer which does the brunt of the work; the deliberator is under the control of the sequencer. In fact, the deliberator's output is merely used as advice by the sequencer, and the entire system is able to function without the deliberator, if necessary. In the DSN automation architecture, as mentioned above, scheduling occurs hierarchically before planning; both steps are necessary. Also, there is a control and execution tier, which is separate from the scheduling tier, unlike ATLANTIS, which combines sequencing with control.

TCA [18] has no real tiers, but many distributed modules working with a central control module via message passing. There is no hierarchy that sets up schedules or plans; TCA operates by setting up a task tree instead.

AuRA [1, 2] has three-tiers: planning, sequencing, and execution for use in mobile robot navigation. Its sequencer simply traverses a FSA expression of a plan, unlike the more powerful algorithms used for scheduling in DANS/OMP. Also, AuRA first plans and then sequences, whereas the DSN automation architecture first schedules, then plans.

The Cypress [19] architecture has plan and execution modules, which operate asynchronously. There is also an uncertainty-reasoning module that communicates with both the plan and execution modules. The DSN Automation architecture's scheduling, plan and execution modules can operate asynchronously, but there is no separate uncertainty-reasoning module. Each tier handles uncertainty independently. Cypress is also not truly a hierarchical architecture and has no scheduling component. The military domain that Cypress has been used for is fairly complex, but since there is no scheduling component, Cypress doesn't tackle as comprehensive a problem as that described in this paper.

Both SOAR [16] and Guardian [13] are general reasoning systems that can be adapted to a given task environment. The algorithms of the planner and the scheduler in the DSN automation architecture could be applied to a number of domains. The execution tier in our architecture, though, is particular to the antenna operations domain. Guardian does not have a hierarchical architecture, but uses a blackboard architecture with one module devoted to scheduling, planning, and control. SOAR also collapses all the tiers into a single mechanism.

The DSN automation architecture uniquely combines a scheduler, planner, and execution module to automate a complex domain with many conflicting, hard constraints, handling re-planning and rescheduling as necessary. The systems which have been designed for mobile robot navigation do not operate in as complex a domain as the DSN antenna operations domain. Examining the general reasoning systems, these are not hierarchically organized into separate planning, scheduling, and execution tiers. This hierarchical organization is a necessary part of the DSN antenna operations domain. The DANS/OMP scheduler uses more powerful algorithms then any of the other described systems' schedulers or sequencers. Unlike most of these systems, in the DSN antenna operations domain, it is necessary to first schedule and then plan, rather than plan and then schedule. Lastly, during execution, none of the other systems described appear to be capable of communicating with as large a set of external equipment as there are in the DSN antenna operations domain, monitoring for possibly multiple antenna or subsystem failures.

10. Insights To DSN Automation

In the past, the process of operating DSN antenna stations, and providing communications passes has been labor and knowledge intensive. Until very recently, automation in the

DSN was little more than a terminal connected to a network monitoring the status of the system through the manual inspection of monitor data values, and a command line interface for executing directives. Things improved when macro scripting was developed to enable operators to create macros (i.e. activity sequences) to perform many of the commonly performed tasks. Still this required large amounts of operator expertise and knowledge. In this mode, an operator monitors each communications pass, where as in the future (near term) design one operator would monitor a number of passes.

Recently, efforts have been made to reduce the cost of operations [14]. One such effort has been in the area of automation. Many approaches have been applied to automation control / commanding of different types of systems. In the AI group at JPL we have worked on automating the generation of control / command sequences, which can be run as control scripts to operate the station. This process was described in the DS-T script generator portion of this paper. Besides the current work presented in the discussion of DS-T many other efforts have and are focussing on the future of DSN automation. Here we will give a brief overview of some of the work and technologies being applied to further increase productivity, as the demand for service goes up and the realities of economics require a reduction in operating cost.

At JPL there has been a large amount of work in addressing these issues. One ongoing effort has been the Network Control Project (NCP), which is made up of two smaller efforts:

- Network Planning Preparation (NPP), and
- Network Monitor and Control (NMC).

The NPP effort addresses issues of infrastructure. This element includes: schedule generation, service request generation, predict generation among others. Many of these functions are interdependent upon each other and in the past have involved a large manual component. Through the automation of these support functions an environment will be created enabling much more complete automation of other aspects of the process. Much of the benefit will come through the sharing of data and the transfer of products from one phase to another, where each phase performs a particular aspect needed to provide the overall service.

The NMC project element provides real-time monitor and control of the DSN. The DSN's intelligent controllers receive directives from the NMC and, in turn, issue commands to the transmitters, receivers, and other subsystems. The primary focus of this work has been in the development of a facility for capturing operator knowledge and desires, and converting them to system control directives (instructions). This has consisted of the capture of domain knowledge in the form of control scripts[2] and the development of a framework for the execution of this control scripts. A large portion of the frame work addresses issues such as how to share data among different components of the system, and providing interfaces for the operators to interact with system. To date the NMC has automated the pre-track and post-track portion of a communications pass, but not the track segment itself. Automation of the track phase of a pass is currently being considered and offers great potential benefit.

One of the least automated components of the NPP charter is the area of resource allocation/scheduling. Due to the nature of the limited resources for communication service, the allocation of these resources, primarily the antennas, is an important process. The resource allocation process (RAP) begins at the inception of a new mission to forecast the needs of that mission and influence its communication design. This process continues throughout the life of the mission. Currently this RAP process is a very labor-intensive process involving a lot of negotiating. Automated scheduling is one technique being applied to both reduce the cost of producing an antenna allocation schedule and to also improve the quality of the schedule through denser use of the resources.

Due to the legacy system complexities in automating the DSN, another task was started, the Network Simplification Project (NSP). The NSP charter has been in guiding future development of the DSN. Part of the findings of the NSP pointed to the need to update many of the legacy systems to provide common interfaces and to use common protocols. One of the driving factors behind these interface upgrades is to enable greater levels of automation in the realm of monitor and control. Much of the NSP work is focussed on adopting many of the concepts demonstrated. Most significantly the station centric approach.

Looking further into the future of the DSN, many forms of advanced technologies are being developed and prototyped. Currently in the AI group at JPL, we are working on modifying and extending the current ASPEN Track Plan Generator to provide a Closed Loop Error Recovery system (CLEaR) for DSN track automation. CLEaR is a real-time planning system built as an extension to ASPEN [8]. The approach taken is to dynamically feed monitor data (sensor updates) back into the planning system as state updates. As these dynamic updates come in, the planning system verifies the validity of the current plan. If a violation is found in the plan, the system will perform local modification to construct a new valid plan. Through this continual planning approach, the plan is disrupted as little as possible and the system is much more responsive and reactive to changes in the real (dynamic) world. The CLEaR effort is being

[2] In the NMC, control scripts are referred to as Temporal Dependency Networks (TDNs).

integrated with a Fault Detection, Isolation and Recovery (FDIR) system.

The larger encompassing task, FDIR, consists of a number of different tools:

- For the detection phase, a system called Beacon-based Exception Analysis for Multi-missions (BEAM) will be used.
- For the isolation phase, of the system Spacecraft Health INference Engine (SHINE) will be used [15].
- For the recovery phase, of the FDIR process CLEaR will be used.

BEAM is a *system level* reasoning component that detects shifts in the physical behavior and trends within a system. In doing so BEAM produces maps that can be graphically depicted to indicate the channels (sensor/monitor data) resulting in the fundamental change to the systems dynamics. From these maps diagnostics information can be derived and fed back into the recovery system. In this fashion BEAM also aids in the isolation phase of the overall system.

SHINE is a reusable inference engine for the monitoring, analysis and diagnosis of real-time and non-real-time systems. It is intended for those areas where inference speed, portability and reuse are of critical importance. The SHINE expert system implemented in the FDIR task provides *monitor data* analysis. SHINE applies a series of rules once an error has been detected, in order to diagnose/isolate the source of the problem. This too is fed back into the recovery system for further analysis.

As part of the FDIR tools set, CLEaR simulates the execution of its plan using monitor data to update its internal state representation. When a state update causes a change from the forecasted state of the system, CLEaR replans in order to correct its internal representation. In many cases the updates may cause as little change as altering the start and end times of upcoming activities within the plan, but in other instances the activities within the plan may be altered or resequenced. It is in this case that CLEaR provides feed back to the primary control system. This feedback is in the form of *recovery sequences* (activities), thus providing the *error recovery* information.

As is often the case, system monitor (sensor) data is often related in different ways that becomes difficult for humans to detect. The advantage of combining these systems together, is that FDIR can first interpret the vast amount of data, summarize it into a set of meaningful values, and provide this as a input for a planning system to react to in providing a recovery plan. We think of this union as intelligent analysis and intelligent response, much like a careful design and implementation; one without the other is of little use.

11. Conclusions

This paper has described an architecture for an autonomous deep space tracking station, DS-T, and offered insights to future areas of DSN automation. This DS-T station automates routine operations such as: scheduling and resource allocation, antenna and receiver predict generation, track procedure generation from service requests, and closed loop control and error recovery for the station subsystems. This architecture has been successfully demonstrated through a set of DS-T technology demonstrations.

12. Acknowledgements

Portions of this work were performed by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration. Other portions of this work were performed by Integral Systems Incorporated under JPL Contract. This task involved a large team and the authors would like to acknowledge the efforts of the entire DS-T team and the support received from the personnel at the Goldstone signal processing complex.

References

[1] R. Arkin. Motor schema-based mobile robot navigation. *International Journal of Robotics Research*, 8(4), 1989.

[2] R. Arkin and T. Balch. AuRA: Principles and Practice in Review. to appear in *Journal of Experimental and Theoretical Artificial Intelligence*, Vol. 9, No. 2, 1997.

[3] R.P. Bonasso, R.J. Firby, E. Gat, D. Kortenkamp, D.P. Miller, and M.G. Slack. Experiences with an Architecture for Intelligent, Reactive Agents. to appear in *Journal of Experimental and Theoretical Artificial Intelligence*, March 1997.

[4] C. Borden, Y.-F. Wang, and G. Fox, “Planning and Scheduling User Services for NASA’s Deep Space Network,” *Working Notes of the 1997 International Workshop on Planning and Scheduling for Space Exploration and Science*, Oxnard, CA, 1997.

[5] R. Brooks, “A Robust Layered Control System for a Mobile Robot,” *IEEE Journal of Robotics and Automation*, 2(1), 1986.

[6] S. A. Chien, R. W. Hill, Jr., A. Govindjee, X. Wang, T. Estlin, M. A. Criesel, R. Lam, and K. Fayyad, “A Hierarchical Architecture for Resource Allocation, Plan Execution, and Revision for Operations of a Network of Communications Antennas,” *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*, Albequerque, NM, 1997.

[7] S. Chien, F. Fisher, H. Mortensen, E. Lo, R. Greeley, A. Govindjee, T. Estlin, X. Wang, "Using Artificial Intelligence Planning Techniques to Automatically Reconfigure Software Modules," 11th International Conference on Industrial and Engineering Application of Artificial Intelligence and Expert Systems, Castellon, Spain, June 1998.

[8] S. Chien, R. Sherwood, R. Knight, A. Stechert, and G. Rabideau, "Integrated Planning and Execution for Autonomous Spacecraft," To appear in the *Proceedings of the 1999 Space Technology and Applications International Forum*, Albuquerque, NM, January-February, 1999.

[9] T. Estlin, F. Fisher, D. Mutz, S. Chien, "Automated Planning for A Deep Space Communcation Station," To appear in the *Proceedings of the 1999 IEEE Aerospace Conference*, Aspen, CO, March 1999.

[10] A. Fukanaga, G. Rabideau, S. Chien, D. Yan, "Toward an Application Framework for Automated Planning and Scheduling," *Proceedings of the 1997 International Symposium of Artificial Intelligence, Robotics and Automation for Space*, Tokyo, Japan, July 1997.

[11] R.J. Firby. Adaptive Execution in Complex Dynamic Worlds. PhD Thesis, Yale University, 1989.

[12] E. Gat. Integrating planning and reacting in a heterogeneous asynchronous architecture for controlling real-world mobile robots. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 1992.

[13] B.Hayes-Roth. An Architecture for Adaptive Intelligent Systems. *Artificial Intelligence*, 72, 1995.

[14] R. W. Hill, Jr., S. A. Chien, and K. V. Fayyad, "Automating Operations for a Network of Communications Antennas," *Proceedings of the 1996 IASTED International Conference on Artificial Intelligence, Expert Systems, and Neural Networks*, Honolulu, HI, August 1996.

[15] M. L. James, and D. J. Atkinson, "Software for Development of Expert Systems," *NASA Tech Brief* Vol. 14, No. 6, Item #8 from JPL Invention Report NPO-17536/7049 June 1990

[16] J.E. Laird, A. Newell, and P.S. Rosenbloom. SOAR: An Architecture for General Intelligence. *Artificial Intelligence*, 33(1), 1987.

[17] D.J. Musliner, E. Durfee, and K. Shin. CIRCA: A cooperative, intelligent, real-time control architecture. *IEEE Transactions on Systems, Man and Cybernetics*, 23(6), 1993.

[18] R. Simmons. Structured Control for Autonomous Robots. *IEEE Transactions on Robotics and Automation*, Vol. 10, No. 1, February 1994.

[19] D.E. Wilkins, K.L. Myers, J.D. Lowrance, and L.P. Wesley. Planning and Reacting in Uncertain and Dynamic Environments. *Journal of Experimental and Theoretical Artificial Intelligence*, 7, 1995.

BIOGRAPHIES

Forest Fisher is a member of technical staff in the Artificial Intelligence Group of the Jet Propulsion Laboratory, California Institute of Technology where he performs research and development of automated planning and scheduling systems for science data analysis and ground station automation. He is also the software lead and the automation lead for the DS-T task. He holds a B.S. in Computer Science from the University of Texas, and is currently completing a M.S. in Computer Science at the University of Southern California. His research interests are in the areas of: planning and scheduling, operations research, monitor and control, and signal processing, and is currently doing work in autonomous control systems and resource scheduling for NASA deep space communications.

Darren Mutz *is a member of technical staff in the Artificial Intelligence Group of the Jet Propulsion Laboratory, California Institute of Technology where he performs research and development of automated planning and scheduling systems. He holds a B.S. in Computer Science from the University of California Santa Barbara.*

Tara Estlin is a member of the Artificial Intelligence Group at the Jet Propulsion Laboratory in Pasadena, California where she performs research and development of planning and scheduling systems for rover automation and ground station scheduling. She received a B.S. in computer science from Tulane University in 1992, an M.S. in computer science from the University of Texas at Austin in 1994, and a Ph.D. in computer science from the University of Texas at Austin in 1997. Her current research interests are in the areas of planning, scheduling and machine learning.

Steve Chien *is Technical Group Supervisor of the Artificial Intelligence Group of the Jet Propulsion Laboratory, California Institute of Technology where he leads efforts in research and development of automated planning and scheduling systems for science data analysis, ground station automation, and highly autonomous spacecraft. He is also an adjunct assistant professor in the Department of Computer Science at the University of Southern California. He holds a B.S., M.S., and Ph.D. in Computer Science from the University of Illinois. His research interests are in the areas of: planning and scheduling, operations research, and machine learning and he has published numerous articles in* *these areas. In 1995 he received the Lew Allen Award for Excellence and in 1997 he received a NASA Exceptional Achievement Medal both for his research and engineering work in automated planning and scheduling systems.*

Leslie Paal *is a member of technical staff in the Advanced Communications Concepts Group of the Jet Propulsion Laboratory, California Institute of Technology where he does research and development of automated, autonomous Deep Space ground stations. He is the task lead and systems engineer for the DS-T task. He holds a B.S, in Electrical Engineering and Computer Science.*

Emily Law *is a senior software engineer in the Telecommunication Signal Processing Engineering Group of the Jet Propulsion Laboratory, California Institute of Technology where she leads efforts in software architecture and development of the Block V Receiver. She was also the software lead for the DS-T task. She holds a B.S. in Math/Computer Science from the California State University and a M.S. in Computer Science from the University of Southern California.*

Nasser Golshan *is a member of the Advanced Communications Concepts Group of the Jet Propulsion Laboratory, California Institute of Technology where he was the task lead for the DS-T task. He holds a Ph.D. in Electrical and Electronic Engineering from the University of Illinois, an M.S. from Seattle University, and a B.A. from The American University of Beirut.*